Workshop on Integration of Vision and Inertial Sensors
Proceedings of ICAR 2003
The 11th International Conference on Advanced Robotics
Coimbra, Portugal, June 30 - July 3, 2003

Fusing visual contour tracking with inertial sensing to recover robot egomotion

Guillem Alenyà
Robosoft
Teknogunea Izarbel
64210 Bidart, France
guillem_alenya@coeic.org

Elisa Martínez
CiTS. Ramon Llull University
Pge. Bonanova 8
08022 Barcelona
elisa@salleurl.edu

Carme Torras
Institut de Robòtica (CSIC-UPC)
Llorens i Artigues 2-4
08028 Barcelona
ctorras@iri.upc.es

Abstract

A method for estimating mobile robot egomotion is presented, which relies on tracking contours in real-time images acquired with an uncalibrated monocular video system. After fitting an active contour to an object in the image, 3D motion is derived from the affine deformations suffered by the contour in an image sequence. More than one object can be tracked at the same time yielding some different pose estimations. Then, improvements in pose determination are archieved by fusing all these different estimations. Inertial information is used to obtain better estimates, as it introduces in the tracking algorithm a measure of the real velocity. Inertial information is also used to eliminate some ambiguities arising from the use of a monocular image sequence. As the algorithms developed are intended to be used in real-time control systems, considerations on computation costs are taken into account.

1 Introduction

Points have been the traditional source of information for image-based tracking algorithms. More complex structures such as lines and corners have also been used. Recently, active contours fitted to image objects have been proposed as tracking features [7], especially in applications not requiring a high precision, as it is the case of robot navigation. Tracking active contours provides an estimate of robot position and orientation ("pose", for short), by relating it to the changes of the contour projection in the image plane. A monocular vision system is used and the image projection is modelled using the weak-perspective camera model.

The algorithm for pose recovery has the following four steps. First, the active contour has to be initialised. One common way to represent active contours is by using b-splines [1]. In this work, initialisation of the b-spline is manually performed by an operator. When corners are present, the use of a corner detector [5] improves the initial adjustment. Automatic initialisation techniques have been proposed [3] and tested with good results. Since we are assuming weak perspective, only affine deformations of the initial contour will be allowed by the tracker and, therefore, the initialisation process is important as it determines the family of affine deformations that the contour will be allowed to adjust to.

Second, a tracking algorithm recognises the new position of the contour in the image and computes the change in position. A Kalman filter is used to integrate this new measurement with the history of previous ones. Since an affine deformation of a b-spline can be parameterised using a shape vector [2], this is used as the state vector for the filter. The shape vector is a set of 6 parameters representing the deformation of the b-spline independently of the complexity and the number of control points of the b-spline used. Tracking one contour provides an estimation of its shape vector S and its associated covariance matrix Γ_S.

The tracking strategy is here enriched by using information provided by inertial sensors. Tracking at low velocities is known to be a favourable case, while tracking usually fails at high velocity rates. Using the good conditioning of inertial sensors precisely to detect relative high velocities, the tracking algorithm is improved by introducing the measured dynamics.

Third, from the shape vector S, the 3D camera pose is computed. It has been proved [6] that it is possible to obtain rotation and tranlation information from the shape vector by using SVD decomposition techniques. Using these techniques, the difference between the initial and the current pose is computed. It is worth noting that this is not an incremental method and, consequently, it does not suffer from the typical accumulative bias produced by signal integration in odometry and inertial-based navigation systems.

Finally, by fusing the information provided by the several contours available in the image, the 3D robot pose is obtained. Rotations are absolute values, whereas translations are recovered up to the scaled depth. Inertial measurements are useful also here to

solve some ambiguities in the result arising from the use of a monocular vision system.

The paper is structured as follows. The next section makes some considerations to be taken into account in the redesign of the tracking algorithms to allow for the simultaneous tracking of several active contours. Section 3 gives the main ideas to understand the tracking of one contour and how inertial data is used to improve tracking, and the next section explains how to extract 3D pose from tracked contours. Section 5 describes how inertial information helps to solve some ambiguities of the recovered pose. The results of two experiments are presented in Section 6 and, finally, some conclusions and the envisaged future work are discussed in Section 7.

2 From one to n trackers

One tracker provides, in principle, enough information to obtain an estimate of the change in robot pose from the initial frame. Although the tracking algorithm is designed to be robust to some kinds of partial occlusions, there are some situations where the tracker gets necessarily lost and fails to recover the true 3D pose. For example, under an automatic initialisation scheme, a contour may be fitted to a *moving* object. In this case, the estimate would not be the robot movement, but the relative motion between the moving object and the vehicle (note that, if the camera is known not to be moving, then the object movement is recovered). An active contour can also be initialised to fit a very close object, or a robot trajectory may take objects closer to the camera. In such situations, perspective effects appear. Affine transformations are then not enough to encode the deformations of the contours in the image and errors in the recovered pose become larger. Moreover, due to the camera model adopted, the depths of the points that form the contour must be small, as the contour is taken to be a planar one. Effects of apparent contours[4] in the image may produce large errors in estimates.

By tracking simultaneously more than one contour in the image, several pose estimates are obtained. It is expected that, by fusing all these estimates, improvements in both pose estimation and tracking robustness will be obtained.

Since all contours deform as a result of the same camera motion, one could easily think of sharing the same Kalman filter state between all contours. However, taking into account the differences in initialisation, as well as the disruptive phenomena explained above, it turns out that this is not a good option. Trackers should be as independent as possible to provide separate estimates, most of which will hopefully be free of the disruptions in image projection mentioned above. In this way, outliers could be spotted and a reliable estimate could be derived by fusing the estimates from the remaining trackers.

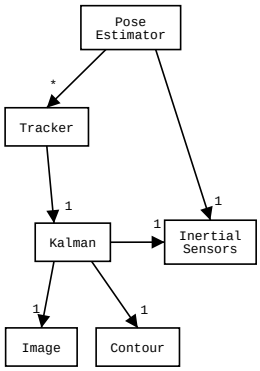


Figure 1: *Software design allowing the tracking of several contours.*

Following this idea, the software design is shown in Figure 1. Each tracker can be seen as a different sensor.

3 Tracking strategy

The objective of tracking is to follow an object contour along a sequence of images. Due to its representation as a b-spline, the contour is divided naturally into sections, each one between two consecutive nodes. For the tracking, some interest points are defined along each contour section. Passing through each point and normal to the contour, a line segment is defined as shown in Figure 2. The search for edge elements (called "edgels") is performed only for the pixels under these normal segments, and the result is the Kalman measurement step. This allows the system to be quick, since only local image processing is carried out, avoiding the use of high-cost image segmentation algorithms.

Once edge elements along all search segments are located, the Kalman filter fuses this measured contour with that predicted from previous history, so that the resulting shape vector is always an affine deformation of the initial contour.

The length of the search segments is determined by the covariance estimated in the preceding frame by the Kalman filter. This is done by projecting the covariance matrix into the line normal to the contour at the given point. If tracking is finding good affine transformations that explain changes in the image, the covariance decreases and the search segments shrank. On the one hand, this is a good strategy as features are searched more locally and noise in image affects less the system. But, on the other hand, this solution is not the best for tracking large changes in image projection.

Large changes in image position can be produced by quick movements of the camera, but also by slow

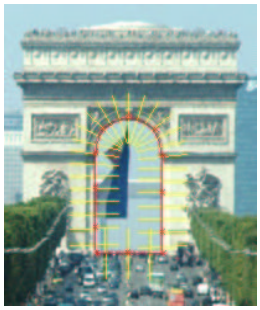


Figure 2: *An image with a contour and its search segments highlighted.*

movements when the object is close to the image plane. As mentioned above, a weak-perspective model is used for camera modelling. To fit the model, the camera field-of-view has to be narrow. In such a situation, distant objects may produce important changes in the image also in the case of small movements of the camera.

For each search segment normal to the contour, the scale factor is computed as:

$$SearchScale = \sqrt{N^T(HPH^T)N}$$

where N are the normal line coordinates, H is the measurement vector and P is the 6×6 top corner of the covariance matrix. Detailed information can be found in [2].

Note that, as covariance is changing at every frame, the search scale has to be recalculated also for each frame. It is also worth noting that this technique produces different search scales depending on the orientation of the normal, taking into account the directional estimation of covariance of the Kalman filter.

Inertial information is added by multiplying the current covariance submatrix by a matrix representing systematic deviation.

$$P_i = P * V$$

$$SearchScale = \sqrt{N^T(HP_iH^T)N}$$

where V is the measurement vector for the inertial sensing system.

In the case of having sensed velocities in a mobile robot, as in our experiments, the matrix representing sensed data with the change of reference frame is of the form:

$$V = \begin{bmatrix} 1 & & .. & & & 1 \\ & v_y & & & & \\ . & & v_x & & & . \\ . & & & v_\omega & & . \\ & & & & 1 & \\ 1 & & .. & & & 1 \end{bmatrix}$$

As will be shown in the experiments section, the tracking algorithm improves when using data from inertial sensors to scale the computed covariance.

4 Obtaining 3D pose from shape vector

Under weak-perspective conditions, the rigid motion of a 3D contour $\mathbf{D}(s)$ relative to a reference contour $\mathbf{D_0}(s)$,

$$\mathbf{D}(s) = R\mathbf{D_0}(s) + \mathbf{T},$$

projects on the camera as an affine deformation of the projection of the reference contour, namely the template, that can be expressed as:

$$\mathbf{d}(s) - \mathbf{d_0}(s) = (M - I)\mathbf{d_0}(s)$$

where $\mathbf{I}$ is the identity matrix,

$$\mathbf{M} = \frac{Z_0}{R_{33}Z_0 + T_Z} \begin{bmatrix} R_{11} & R_{21} \\ R_{21} & R_{22} \end{bmatrix}, \quad (1)$$

$$\mathbf{t} = \frac{1}{R_{33}Z_0 + T_Z} \begin{bmatrix} Z_0R_{13} + T_x \\ Z_0R_{23} + T_y \end{bmatrix}, \quad (2)$$

$\mathbf{R_{ij}}$ are the elements of the 3D rotation matrix $\mathbf{R}$, $\mathbf{T_i}$ are the elements of the translation vector $\mathbf{T}$ and Z_0 is the distance from the template $\mathbf{D_0}(s)$ to the camera.

Using the Euler notation to represent the rotation matrix,

$$\mathbf{R} = \mathbf{R_z}(\phi)\mathbf{R_x}(\theta)\mathbf{R_z}(\psi), \quad (3)$$

equation (1) can be rewritten as

$$\begin{aligned} \mathbf{M} = & \frac{Z_0}{T_z + R_{33}Z_0}\mathbf{R_z}|_\mathbf{2}(\phi)\mathbf{R_x}|_\mathbf{2}(\theta)\mathbf{R_z}|_\mathbf{2}(\psi) = \\ & = \frac{Z_0}{T_z + R_{33}Z_0}\mathbf{R_z}|_\mathbf{2}(\phi) \begin{bmatrix} 1 & 0 \\ 0 & cos\theta \end{bmatrix} \mathbf{R_z}|_\mathbf{2}(\psi) \quad (4) \end{aligned}$$

and,

$$\mathbf{M}\mathbf{M}^T = \mathbf{R_z}|_\mathbf{2}(\phi) \begin{bmatrix} L & 0 \\ 0 & Lcos^2\theta \end{bmatrix} \mathbf{R_z}|_\mathbf{2}{}^{-1}(\phi) \quad (5)$$

where

$$\mathbf{L} = \left(\frac{Z_0}{T_z + R_{33}Z_0} \right)^2$$

This last equation shows that θ can be computed from the ratio of eigenvalues of $\mathbf{MM^T}$, namely (λ_1, λ_2),

$$cos\theta = \sqrt{\frac{\lambda_2}{\lambda_1}}, \quad (6)$$

where λ_1 is the largest eigenvalue and λ_2 the smallest one. The angle ϕ can be extracted from the eigenvectors of $\mathbf{MM}^T$. The eigenvector $\mathbf{v_1}$ with largest eigenvalue equals the first column of $\mathbf{R_z}|_\mathbf{2}(\phi)$,

$$\mathbf{v_1} = \begin{bmatrix} cos\phi \\ sin\phi \end{bmatrix}.$$

Isolating $\mathbf{R_z}|_{\mathbf{2}}(\psi)$ from equation (4),

$$\mathbf{R_z}|_{\mathbf{2}}(\psi) = (R_{33} + \frac{T_z}{Z_0}) \begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{cos\theta} \end{bmatrix} \mathbf{R_z}|_{\mathbf{2}}(-\phi)\mathbf{M},$$

and observing, from equation (5), that

$$R_{33} + \frac{T_z}{Z_0} = \frac{1}{\sqrt{\lambda_1}},$$

we can find $sin\psi$ and then ψ.

Once the angles ψ, θ, ϕ are known, the rotation matrix $\mathbf{R}$ can be computed as in equation (3).

The scaled translation along Z can be computed as

$$\frac{T_z}{Z_0} = \frac{1}{\sqrt{\lambda_1}} - R_{33}. \quad (7)$$

The rest of the components of the 3D translation vector can be computed from $\mathbf{t}$ and $\mathbf{R}$ using equation (2),

$$\frac{T_x}{Z_0} = \frac{t_x}{f\sqrt{\lambda_1}} - R_{13}, \quad (8)$$

$$\frac{T_y}{Z_0} = \frac{t_y}{f\sqrt{\lambda_1}} - R_{23}. \quad (9)$$

5 Estimating 3D pose

¿From each tracker a shape vector and its covariance is obtained. The system pose estimation is computed using all the individual estimations.

The preceding section has defined a method to translate tracked data from shape vector space into motion parameters in 3D space. One option to combine all these informations is to use statistical methods. Statistical fusion [8] is proposed as a first approach.

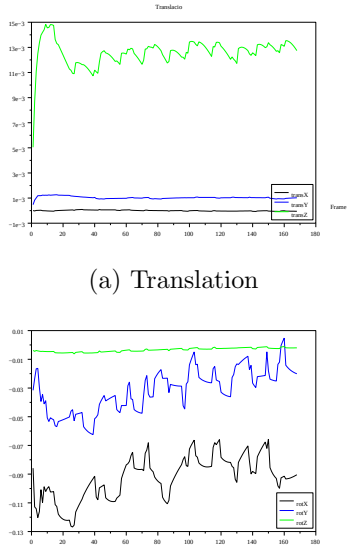
Assuming no outliers in the tracked contours, all the estimates can be combined to obtain a better estimation of the pose. We propose the trace of the covariance matrix as an approximation of the variance of each pose parameter. An individual estimation of pose is obtained from each contour at frame rate.

For simplicity we analyse the case of two estimations. For each degree of freedom $(ti_x, ti_y, ti_z, wi_x, wi_y, wi_z)$, the conditional probability of the new estimated value, for example t_x, is based on estimations $t1_x$ and $t2_x$ from two different trackers, and can be computed using:

$$t_x = \left[\frac{\sigma^2_{t2_x}}{\sigma^2_{t1_x}\sigma^2_{t2_x}}\right] t_1 + \left[\frac{\sigma^2_{t1_x}}{\sigma^2_{t1_x}\sigma^2_{t2_x}}\right] t_2$$

$$\frac{1}{\sigma^2} = \frac{1}{\sigma^2_{t1_x}} + \frac{1}{\sigma^2_{t2_x}}$$

Approximating all components by the same variance value is a gross approximation. It is easy to prove that not all translations and rotations will be sensed with the same resolution. Translations parallel to the image plane are better sensed as they produce greater changes in image than translations normal to the image plane. The same reason explains that rotations about an axis perpendicular to the image plane are better sensed than the ones about axes parallel to the image plane. This can be seen in Figure 3, where a static camera trackes a moving contour. In this experiment a synthetic image is used in order to avoid measuring noise from the acquisition system. As predicted, in translation (Figure 3(a)) depth is worst estimated, appearing greater variance. In rotational recovered parameters (Figure 3(b)), Z varies more than X and Y.



(a) Translation

(b) Rotation

Figure 3: *3D pose evaluated in a free-of-motion contour.*

Using monocular vision one can observe that different real 3D movements result in the same, or very similar, projections on the image plane. This is due to both finite resolution and the well-known projection ambiguities.

One of these ambiguities is the so called Necker reversal ambiguity. It is explained in Figure 4. Take a point rotating a certain angle about a point in a plane parallel to the image plane. Its projection in the image will be very similar to that of another point, in mirror position from the parallel plane, rotating the same angle but in the opposite direction.

When projection effects are big, the sign of the angle is easy to recover. When these effects diminish, as it is the case of this work, the direction becomes undetermined. This can be determined using an inertial

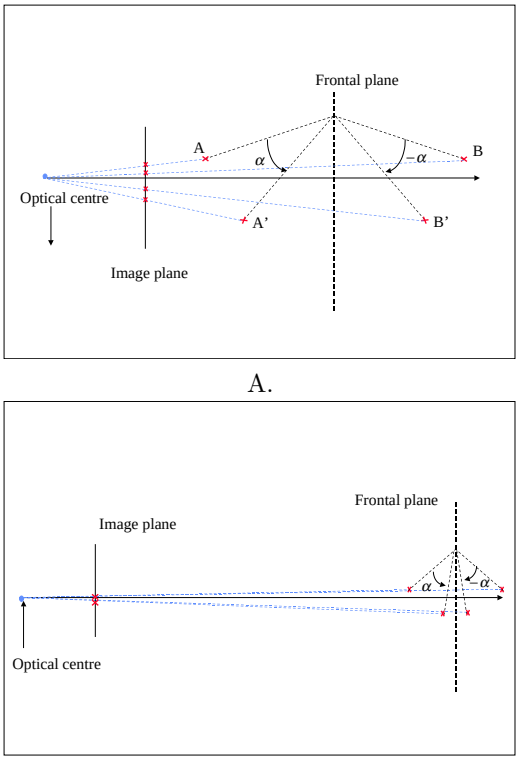


Figure 4: *Necker reversal ambiguity appears when the perspective effects diminish. There is no ambiguity in the upper figure. But there is no difference between the image obtained when A rotates to A' and the image recorded when B rotates to B' in the lower figure.*

sensor, as it provides the sign of the movement.

Other ambiguities should be solved, or at least limited using inertial data. However, it requires more precise and robust data from the inertial sensors, which we have not available at the moment.

6 Experiments and results

6.1 Pose estimation with 2 tracked contours

To evaluate the performance of the tracker fusion, synthetic images are used. In this way, errors produced by the tracking system can be easily isolated and the fusion performance can be better evaluated.

A sequence of 100 frames is used simulating a camera rotating about the Z axis. Each successive frame is rotated by 1 degree, thus the total rotation is 100 degrees.

Different initialisations are used for each contour. As can be seen in figure 5, for defining right contour have been used double number of control points rather than in definition of left one.

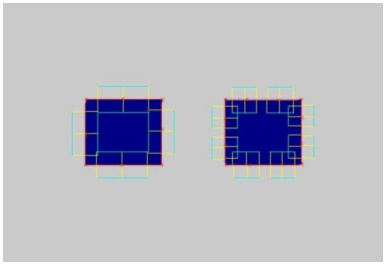


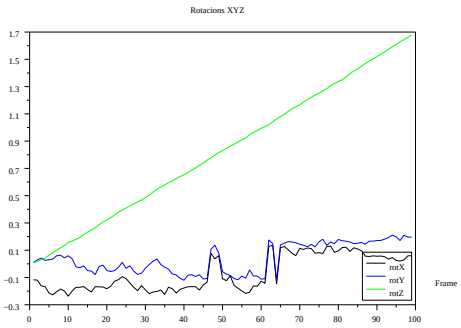
Figure 5: *Different initialisations used in the experiment.*

In figure 6 are shown recovered rotations for both contours. As rotation in ZXZ Euler representation is not intuitive, angles are expressed in XYZ form. Recovered rotation in Z axis is quite well recovered through sequence of frames. In the case of rotations about X and Y axes, just noise is recovered as the camera does not move on these directions.

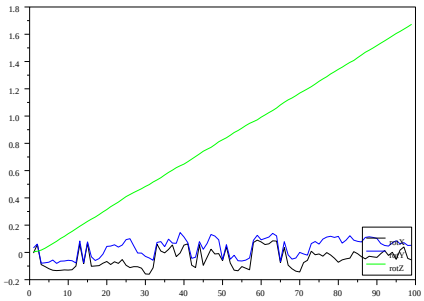
Effects of different initialisations can be seen in Table 1. As more points are used to model the right contour, the standard deviation decreases and becomes lower than in the left contour, as shown in Table 2.

As tracking performance improves, the computed covariance also improves, providing a lower mean value of covariance trace and a lower standard deviation. This can be seen in Table 2, meaning that tracking with more points is more stable.

Using statistical fusion, both estimates can be combined to obtain a better estimate. Results of fusion are shown in Figure 7. As can be seen, values for rotation follow real motion. Noise from single estimations is more stable, and covariace is also stable as the figure is taken at very high zooming rate. This



(a) Left



(b) Right

Figure 6: *Rotation evaluated.*

is important, because it shows that covariance can be used as a quality factor for single pose recovery from the tracking algorithm.

6.2 Improved tracking with inertial information

For this work we use a RobuCab Mobile Robot [9]. It can be used in two modes: car-like navigation and bi-directional driving. As can be seen in Figure 8, it is a relatively big mobile vehicle with capacity for up to four people.

For simplicity of the control system, the car-like driving option is used, but better results should be obtained under bi-direction driving mode as the maximum turning angle would increase. In this vehicle we mount a monocular vision system with the described 6 d.o.f. tracking system. A Gyrostar inertial sensor, from Murata, is used to measure rotations about the Y axis. To measure X and Z linear accelerations, an ADXL dual accelerometer from Analog Devices is used. All these sensors are connected to a dedicated board with an AVR processor used to make A/D conversions, pwm decoding and time integration. It has also a thermometer for thermal data correction. This 'intelligent' sensor provides not only changes in velocity, but also mean velocity and position. Drift, typical in this kind of computations, is reset periodically with the information obtained by fusion of the other sensors. This board shares memory with a MPC555 board, which is connected through a CAN bus to the control and vision processing PC. All the system runs under a real-time Linux kernel in a pentium 233 MHz industrial box. Figure 9 shows the hardware components and their interconnections.

$Left$	Rx	Ry
$Mean:$	-0.0684203	0.0397576
$Std-dev:$	0.1296382	0.1086573
$Right$		
$Mean:$	-0.0434605	0.0356667
$Std-dev:$	0.0676008	0.0679300

Table 1: *Statistics for recovered XYZ rotation.*

	$Left$	$Right$
$Mean:$	20.917314	10.782572
$Std-dev:$	0.0001930	0.0000838

Table 2: *Statistics for covariance*

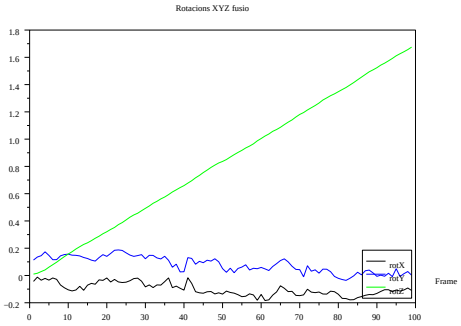
The shape vector we use models all plane deformations resulting from 3D motions. It could be changed in this experiment in order to model only 3 d.o.f. movements[6]. However, as the robot real movements are in 6 parameter space, mainly due to floor rugosity and vehicle dampers, the whole shape vector is used.

In this experiment the robot is in autonomous driving mode, following a filoguided path. In this way, the trajectory can be easily repeated, thus allowing us to perform several experiments with very similar conditions. The path followed consists of a straight line segment, a curve and another straight line.

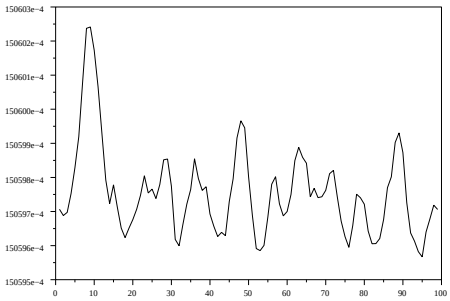
First, algorithm without inertial information is used. On the first straight segment, the contour is well followed, but as can be seen in Figure 10(a), when turning takes place and the contour moves quicker in the image plane, it loses the real object and the covariance increases.

Second, the algorithm adding inertial information to the tracker is used. In this experiment, tracking does not lose the objective and finishes the sequence

	Rx	Ry	Rz	Cov
$Mean:$	-0.10025	0.08014	0.83616	15.0597
$Std-dev:$	0.04602	0.05938	0.49167	0.00013



(a) Rotations



(b) Covariance

Figure 7: *Fusion result.*

giving good recovered pose values. As can be seen in the covariance representation in Figure 10(b), covariance increases at the beginning of the turning, but decreases quickly, showing that tracking has fixed the objective despite its quick translation accross the image.

As can be easily seen, if we use more images trying to track the contour while moving towards it, the contour leaves the image range. Another contour should be initialised in order to be able to continue egomotion recovery in such situation.

7 Conclusions

This article presents a new approach to enrich visual tracking of active contours with information provided by inertial sensors. First, the tracking strategy has been modified to include inertial data. In this way, contour search in the new frame by taking into account its position in the preceding frame is made more robust, thus allowing tracking at higher velocities of the video data.

The paper also presents a framework to extend the method to be able to use multiple contours and shows the way to estimate the 3D pose from several tracked contours. A preliminary approach to fuse different

Figure 8: *Mobile robot used in the experiment.*

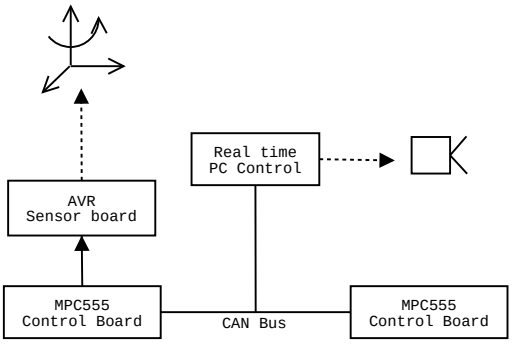


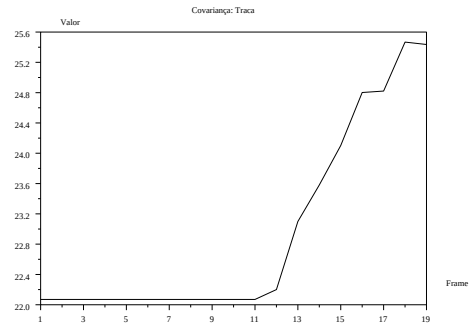
Figure 9: *Interconnection of system sensors and processing units.*

estimations has been presented. Taking advantage of the inertial sensing of the rotation sign, the Necker reversal ambiguity has been solved.

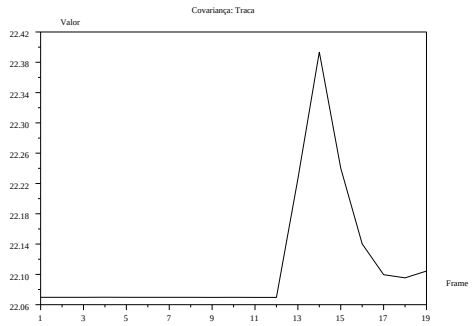
Initial experiments have been performed to prove correctness in some reduced situations. Further work should include the change of the fusion strategy to take into account directional information in the covariance matrix. Improvements in the quality of the data obtained from inertial sensors will allow to reduce, or eliminate some other ambiguitites in the pose recovery algorithm. Experiments using whole information provided by the inertial sensor set should be also performed. As pose recovery is up to 6 dof, experiments with complete 3D movements will be useful to test the feasibility of egomotion recovery algorithms.

Acknowledgments

Guillem Alenyà was supported by the Marie Curie FP5 program. Elisa Martínez and Carme Torras



(a)



(b)

Figure 10: *Covariances resulting from tracking without using inertial information (a) and using it (b).*

acknowledge support from the Spanish Science and Technology Council, project DPI2000-1352-C02-01.

References

[1] D.Reynard A. Blake, M. Isard. Learning to track the visual motion of contours. *J. Artificial Intelligence*, 78:101–134, 1995.

[2] A. Blake and M. Isard. *Active contours*. Springer, 1998.

[3] T. Cham and R. Cipolla. Automated b-spline curve representation incorporating mdl and error-minimizing control point insertion strategies. *IEEE Trans. On Pattern Analisys ans Machine Intelligence*, 21(1), 1999.

[4] R. Cipolla. The visual motion of curves and surfaces. *Phil. Trans. R. Soc. Lond.*, pages 1103–1121, May 1998.

[5] Stephens M. A Harris C. Combined corner and edge detector. *Proceedings of 4th Alvey Vision Conference*, pages 147–51, 1988.

[6] E. Martínez. *Recovery of 3D structure and motion from the deformation of an active contour in a sequence of monocular images*. PhD thesis, 2000.

[7] E. Martínez and C. Torras. Qualitative vision for the guidance of legged robots in unstructured environtments. *Pattern recognition*, 34:1585–1599, 2001.

[8] P. Maybeck. *Stochastic Models, Estimation and Control*, volume 1. Academic Press Inc, 1979.

[9] Robosoft. http://www.robosoft.fr.